Foreword

As with other researchers and authors who study the software business and software engineering, I have had many opportunities to learn about free and open source software (FOSS). There is a lot to know, and I am especially pleased to see this volume of essays from MIT Press because it provides so much information—both quantitative and qualitative—on so many aspects of the open source movement. It will answer many questions as well as continue to inspire more research for years to come.

The research in this book is authoritative and thoughtful and offers something for everyone. For example, economists will want to know the motivations of people and companies (such as IBM or Hewlett Packard), who give freely of their time to create or improve a "public good." Not surprisingly, the research indicates that many FOSS developers are motivated both by the creative challenge as well as self-interest, such as enhancing their reputations as programmers, and then take advantage of this effect when searching for jobs. Because both for-profit and nonprofit organizations pay many programmers to work on open source projects, we find there is also some overlap between the free and open source and commercial software worlds.

Management specialists will want to know if there are business models that enable for-profit firms to take advantage of free or open source software. We learn that there are several seemingly viable commercial opportunities, even though open source, in many ways, is the ultimate commoditization of at least some parts of the software products business. The major business opportunities seem to be the hybrid approaches that make money from selling services (such as for system installation and integration, and technical support) and distributing convenient packages that include both free and open source software as well as some commercial utilities or applications. This is the strategy that Red Hat, the poster child

of commercial OSS companies, has followed, and it is finally making money as a distributor and by servicing Linux users.

Social scientists are fascinated by the coordination mechanisms used in open source projects and will learn a lot about how the process works. Computer scientists and software engineers, as well as IT managers, will want to know if open source development methods produce better software than proprietary methods produce. Most of the evidence in this book suggests that the open source methods and tools resemble what we see in the commercial sector and do not themselves result in higher quality. There is good, bad, and average code in *all* software products. Not all open source programmers write neat, elegant interfaces and modules, and then carefully test as well as document their code. Moreover, how many "eyeballs" actually view an average piece of open source code? Not as many as Eric Raymond would have us believe!

After reading the diverse chapters in this book, I remain fascinated but still skeptical about how important open source actually will be in the long run and whether, as a movement, it is raising unwarranted excitement among users as well as entrepreneurs and investors. On the development side, I can sympathize with the frustration of programmers such as Richard Stallman, Linus Torvalds, or Eric Raymond in not being able to improve commercial software and thus determining to write better code that is free and available. Eric Raymond has famously described the open source style of development as similar to a "bazaar," in contrast to top-down, hierarchical design philosophies similar to how the Europeans built cathedrals in the middle ages.

We also know from the history of the mainframe industry, UNIX, and government-sponsored projects that much software has been a free "public good" since the 1950s and that open source-like collaboration has led to many innovations and improvements in software products. But, on the business side, most companies operate to make money and need some guarantee that they can make a return on investment by protecting their intellectual property. To suggest that *all* software should be free and freely available makes no sense. On the other hand, most software requires an iterative style of development, and at least some software is well suited to being written by programmers for other programmers in an open source mode. Increasing numbers of the rest of us can take advantage of this public good when "programmer products" like Linux, Apache, and Send Mail become more widely used or easier to use.

The conclusion I reach from reading this book is that the software world is diverse as well as fascinating in its contrasts. Most likely, software users

will continue to see a comingling of free, open source, and proprietary software products for as far as the eye can see. Open source will force some software products companies to drop their prices or drop out of commercial viability, but other products and companies will appear. The business of selling software products will live on, along with free and open source programs. This is most likely how it will be, and it is how it should be.

Michael Cusumano
Groton and Cambridge, Massachusetts
February 2005